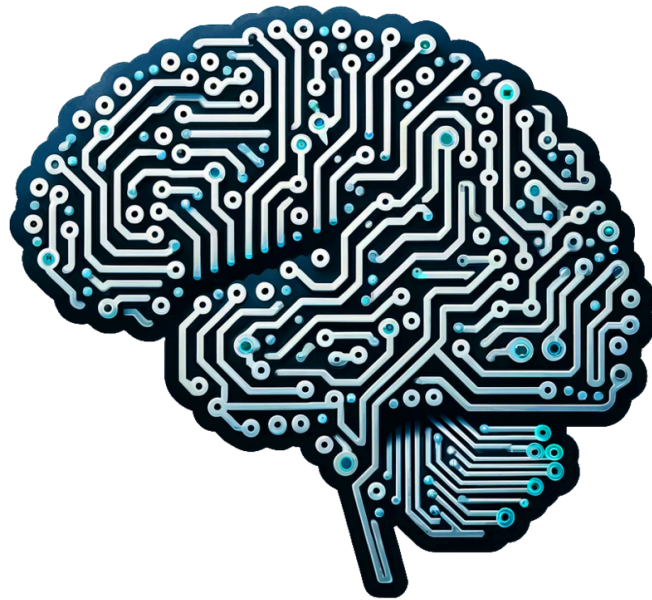


# Introducción a ML y Generative AI



## Taller de programación #1

# Descripción

Este documento contiene información relacionada a las actividades de prácticas del primer taller del curso de introducción a ML y Generative AI . En este taller empezaremos con una pequeña revisión de las herramientas que usaremos en este curso. A continuación mostramos cómo correr instrucciones básicas de Python, instalar librerías externas con pip, y cómo usar un libro de Jupyter notebook dentro de Visual Code. Por favor note que el taller se ha ejecutado en una computadora con el sistema operativo Windows 10.

# 1. Revision

## 1.1 Command Prompt (Simbolo del sistema)

Permite al usuario interactuar directamente con el sistema operativo, los comandos son en forma de línea de textos, antiguamente se usaba para controlar el sistema puesto que no existía mouse, control de manejo de ventanas o GUI.

## 1.2 Editor de código fuente

Es un programa que permite crear o modificar archivos de forma digital. Pueden decidir usar el editor de código fuente de su preferencia para este y los próximos talleres. En caso de no haber usado un editor de código anteriormente les recomendamos Visual Studio Code. Si desean descargarlo pueden hacerlo usando el siguiente enlace: <https://code.visualstudio.com/>.

Nota: Visual Studio Code contiene una terminal integrada, Puedes usar esta terminal si lo deseas una vez hayas instalado Python.

## 1.3 Entorno virtual

Los entornos virtuales son herramientas que permiten a los desarrolladores crear un espacio aislado en el que se pueden instalar las dependencias (a menudo bibliotecas y paquetes) requeridas para un proyecto específico sin interferir con otras instalaciones de software.

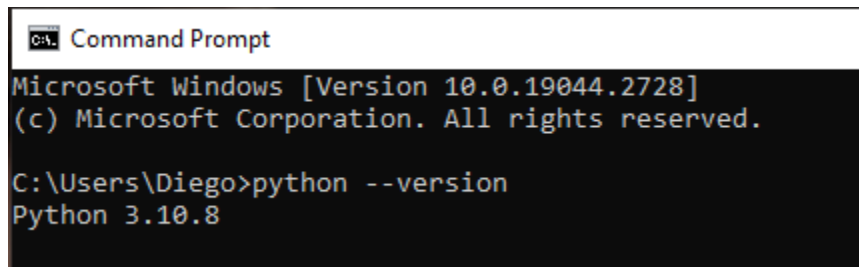
## 1.4 Librerías

Una librería es un conjunto de módulos (código preescrito) que proporciona funciones y clases que puedes reutilizar para resolver problemas comunes sin tener que escribir el código desde cero. Dentro del campo del aprendizaje automático e inteligencia artificial, los científicos y desarrolladores usan librerías como Scikit-learn, TensorFlow, Pandas, entre otras.

## 2. Python

### 2.1 Confirmar que Python está instalado

Recomendamos que confirmes si tienes la versión de Python 3.11, abriendo una terminal o símbolo de sistema e ingresando el comando "python --version" o si hay múltiples versiones de Python instaladas "python3 --version". Si tienes la versión de Python 3.11, puedes ir directamente a la sección 3.



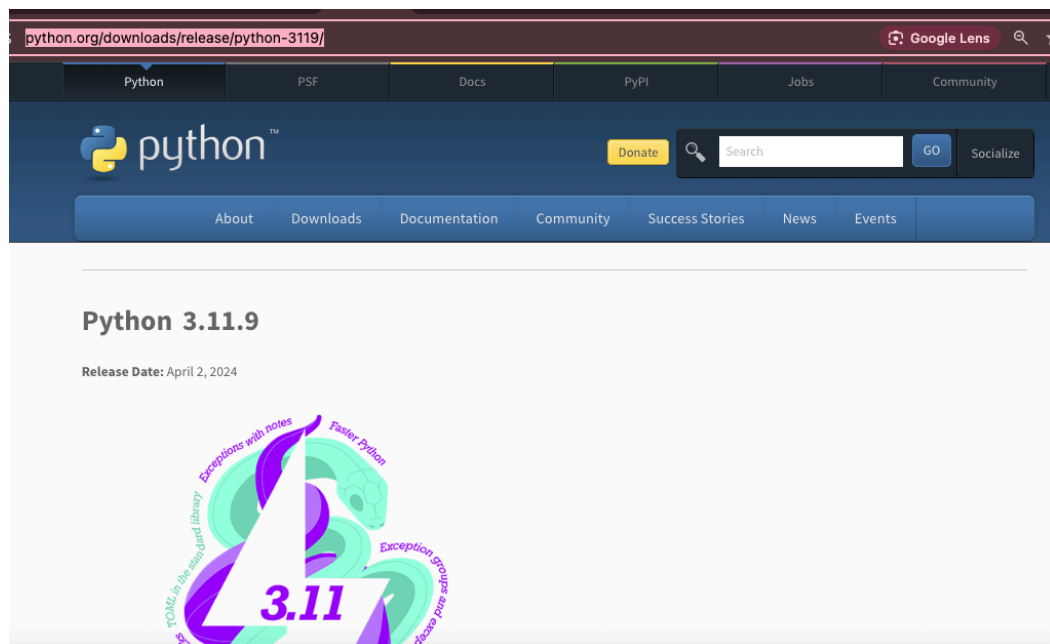
```
Command Prompt
Microsoft Windows [Version 10.0.19044.2728]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Diego>python --version
Python 3.10.8
```

### 2.2 Instalar Python

Para obtener la versión de Python 3.11 usando el siguiente enlace

<https://www.python.org/downloads/release/python-3119/>

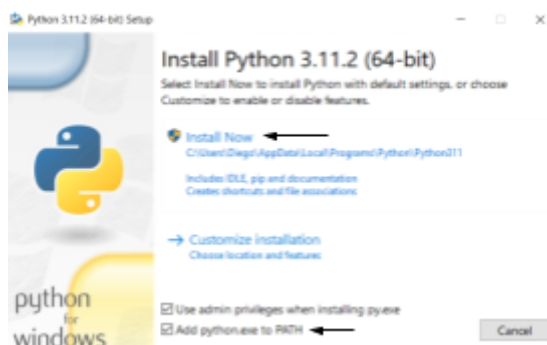


Desciende hasta el final de la página web, dirígete hacia la sección archivos y selecciona el instalador para tu sistema operativo (si estás confundido con el instalador, por favor levanta tu mano durante la hora de taller)

Files						
Version	Operating System	Description	MD5 Sum	File Size	GPG	Sigstore
<a href="#">Gzipped source tarball</a>	Source release		bfd4d3bfeac4216ce35d7a503bf02d5c	25.3 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">XZ compressed source tarball</a>	Source release		22ea467e7d915477152e99d5da856ddc	19.2 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">macOS 64-bit universal2 installer</a>	macOS	for macOS 10.9 and later	fa29f456feb6b5c4f52456a8b8ba347b	42.8 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows installer (64-bit)</a>	Windows	Recommended	e8dcd502e34932eebcdf1be056d5cbcd	25.0 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows installer (32-bit)</a>	Windows		2a1d1ac2d8a0aa847515f9dd121ccbb7	23.8 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows installer (ARM64)</a>	Windows	Experimental	328d93f71cb078965e4cfa2eb2663fa1	24.3 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows embeddable package (64-bit)</a>	Windows		6d9aa08531d48fcc261ba667e2df17c4	10.7 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows embeddable package (32-bit)</a>	Windows		31e7648158376e92a4463aa6f22a78e1	9.6 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows embeddable package (ARM64)</a>	Windows		8611b6aa35483ab1c61d45e0d9f2de0d	10.0 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>

Dependiendo del sistema operativo, deberás descargar diferentes archivos. En el caso que uses un sistema diferente a Windows. Por Favor, envíanos un mensaje para ayudarte con el proceso de instalación.

Para Windows: Una vez que el ejecutable ha sido descargado, es posible iniciar el proceso de instalación haciendo doble clic izquierdo en el ejecutable. Durante la instalación, asegúrate de marcar la casilla "Add python.exe to PATH" y seleccionar "Install Now" para continuar con la instalación.



Una vez el instalador finalice, comprueba que Python está instalado ejecutando los comandos mencionados en la section 2.1

Nota: Es posible que necesites abrir una nueva terminal para comprobar que Python ha sido instalado, o que reinicies tu computador.

## 2.3 Organización de código

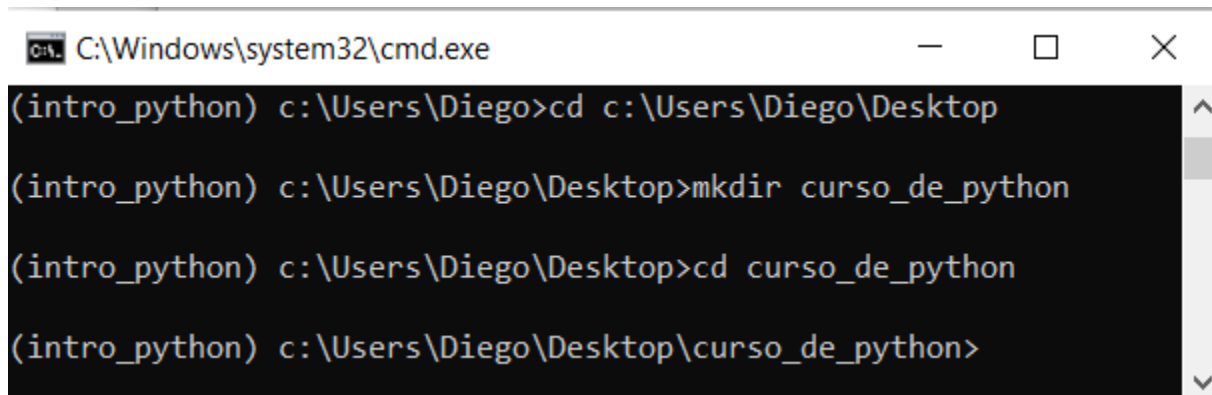
Con el fin de organizar nuestro código, crearemos una carpeta en el escritorio. Para esto ejecute en su símbolo de sistema `> cd c:\Users\(\nombre_de_usuario)\Desktop > mkdir curso_de_python > cd curso_de_python`

## 2.3 Configurar un entorno virtual (Opcional)

Crear un entorno virtual es considerado una buena práctica en el desarrollo de software, ya que evita conflictos de dependencias y problemas de compatibilidad. Python nos permite crear estos entornos usando el siguiente comando: **`python -m venv intro_python`**. Dentro de `curso_de_python` Al hacer esto podrás observar una nueva carpeta en el directorio con el nombre de `intro_python`. Para iniciar el entorno desde la terminal en la carpeta `curso_de_python` usa el siguiente comando:

**Windows:** `intro_python\Scripts\activate.bat`

**Unix or MacO:** `source intro_python/bin/activate`



```
C:\Windows\system32\cmd.exe

(intro_python) c:\Users\Diego>cd c:\Users\Diego\Desktop

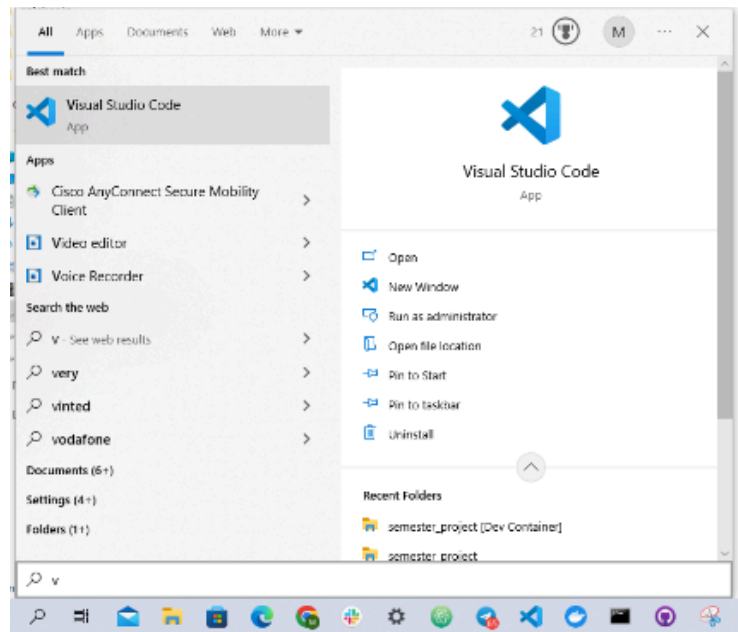
(intro_python) c:\Users\Diego\Desktop>mkdir curso_de_python

(intro_python) c:\Users\Diego\Desktop>cd curso_de_python

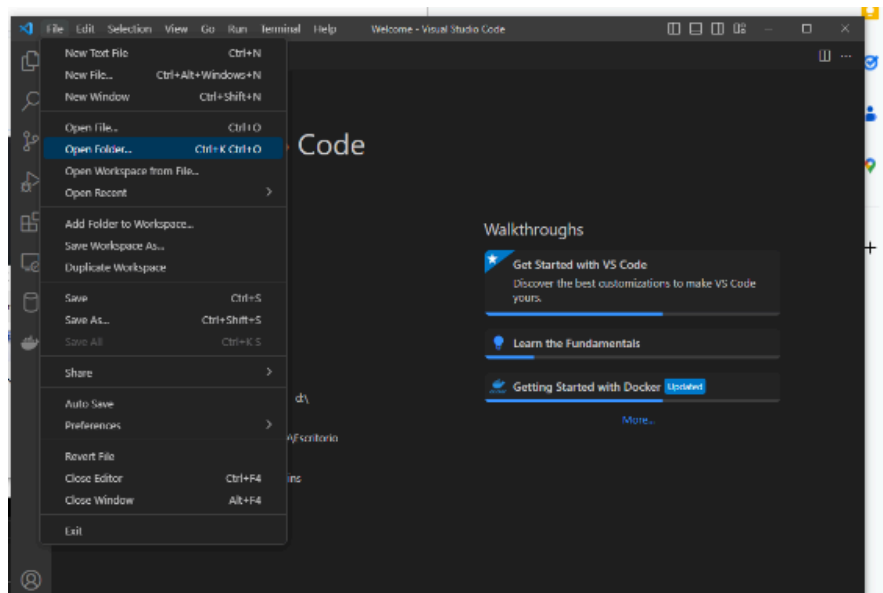
(intro_python) c:\Users\Diego\Desktop\curso_de_python>
```

## 3. Mi primer programa

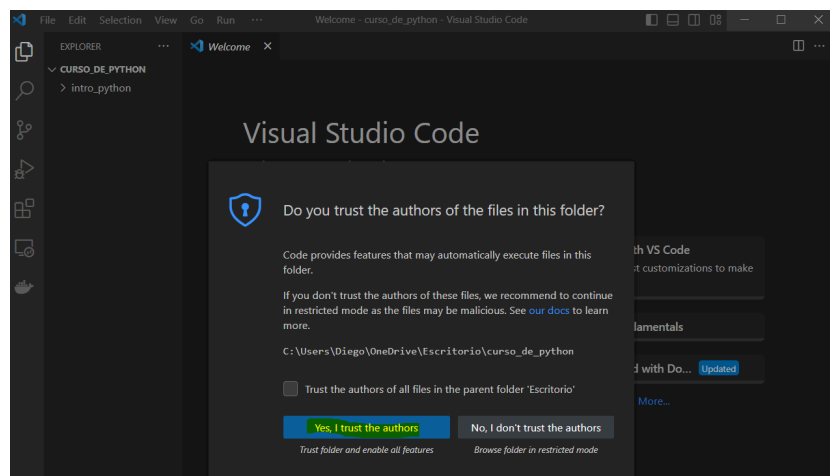
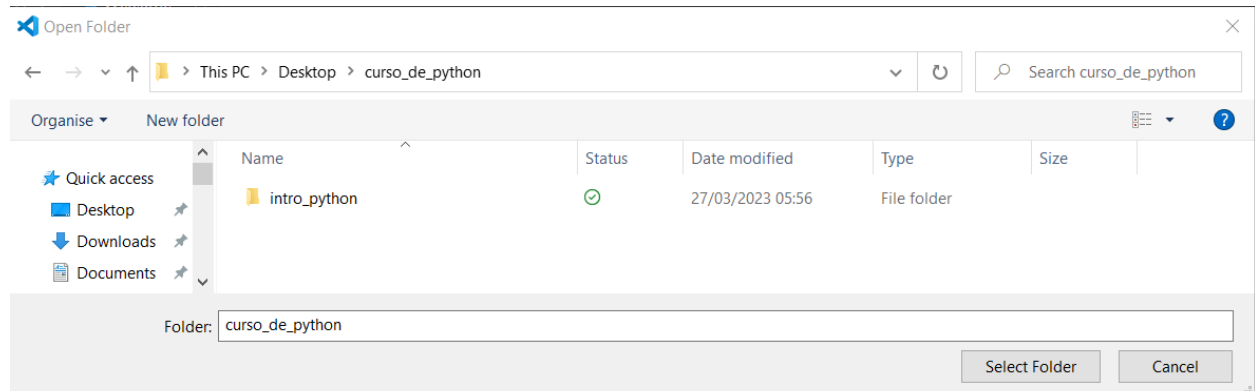
Empezaremos nuestro tutorial abriendo el editor de código fuente “Visual Studio Code” que nos permitirá crear nuestro primer programa. Una vez lo hayas descargado e instalado lo puedes abrir como cualquier otra aplicación de Windows.



Una vez en el editor, podemos abrir la carpeta 'curso\_de\_python' con el material del curso haciendo click en File y luego Open Folder como se muestra en la siguiente imagen:

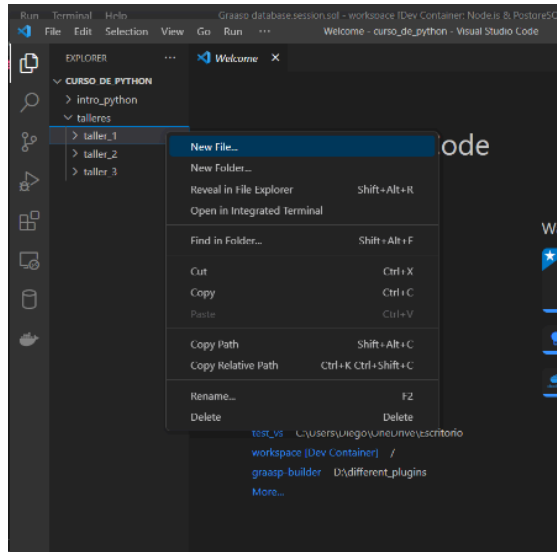


Una vez nos hayamos dirigido a la ruta de nuestra carpeta debemos hacer click en 'seleccionar carpeta' y aceptamos en confiar los autores de la carpeta. Esto nos permitirá visualizar y explorar sus contenidos como se muestra en las siguientes imágenes:

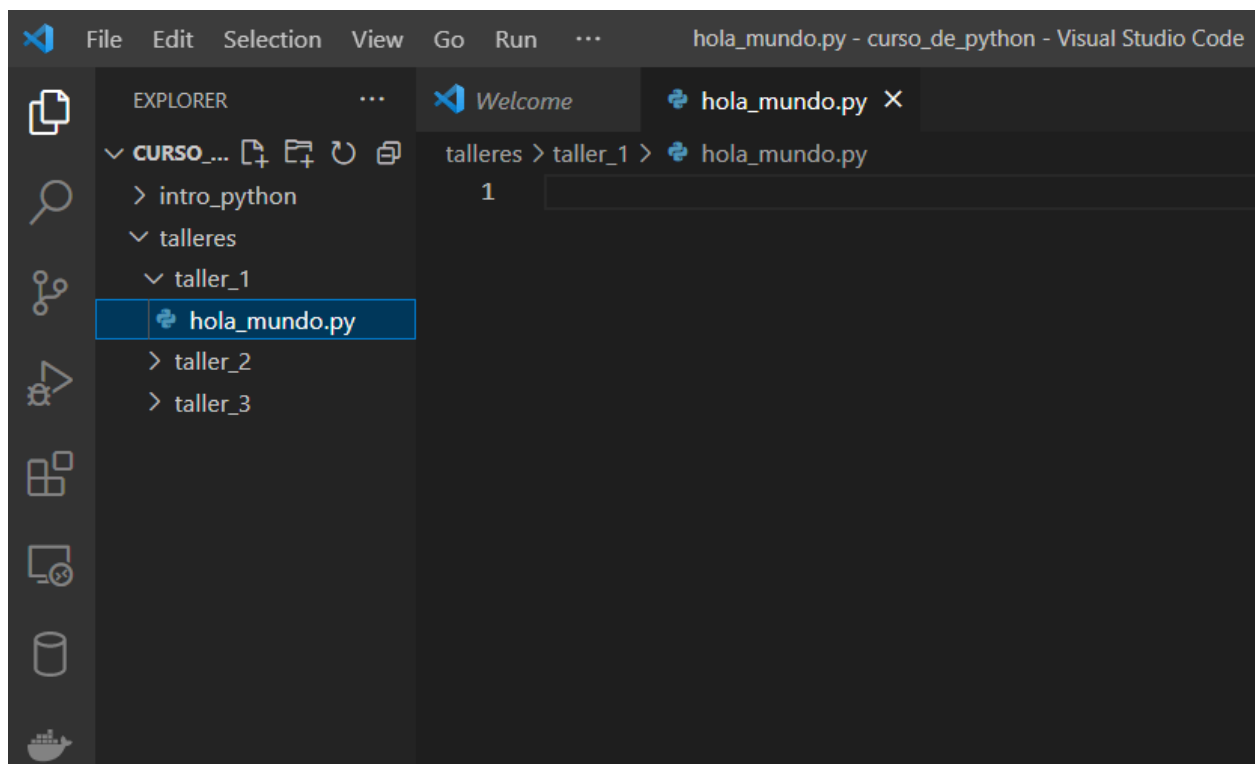


Se puede añadir un nuevo archivo en una carpeta en específico al hacer click derecho sobre esta y seguido seleccionamos 'New File'. En la siguiente imagen se muestra como creamos una nuevo archivo en la carpeta 'taller\_1':



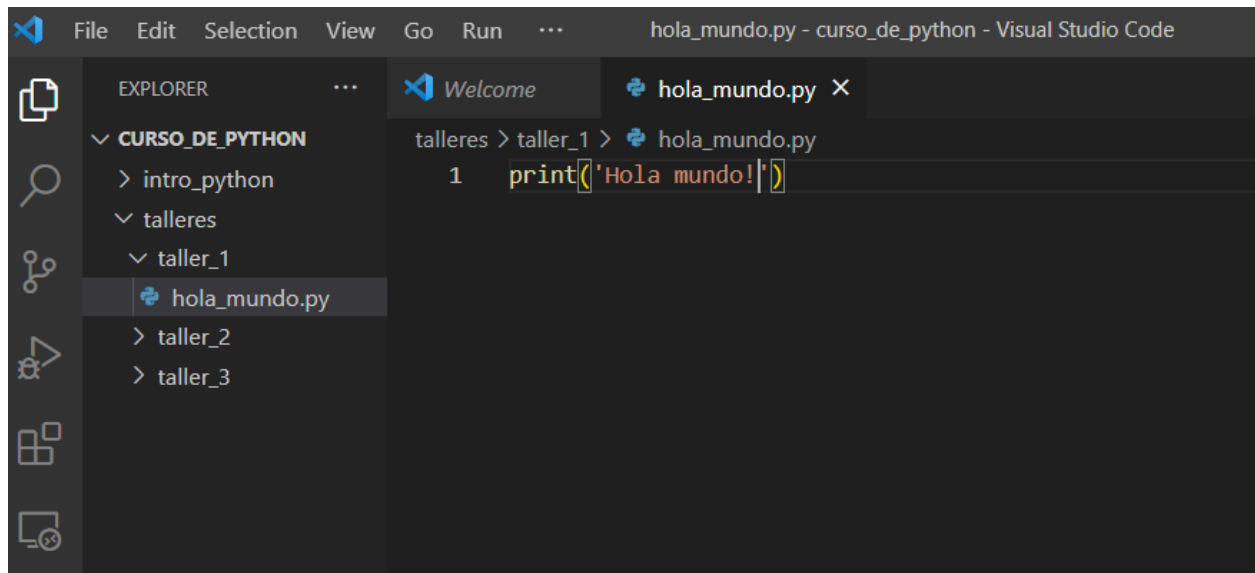


A continuación Visual Studio Code nos preguntara como nombrar al archivo, en nuestro caso lo llamaremos 'hola\_mundo.py'. Recuerda seleccionar la terminación '.py' ya que esto permite que el editor identifique nuestro archivo como un archivo de Python.



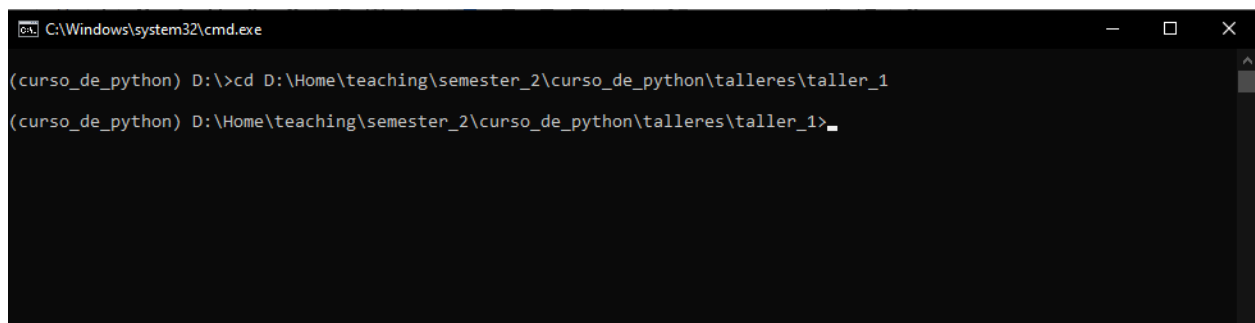
Esto hará que Python cree y abra el nuevo archivo donde podemos escribir sintaxis de Python. Para nuestro primer programa haremos que la consola muestra un mensaje de bienvenida,

para esto usaremos una palabra especial en python llamada print (print nos permite enviar parámetros e imprimirlos linea por linea)



Para guardar los cambios hechos al documento puedes simplemente ocupar el atajo 'CTRL + S' o dirigirte a la pestaña 'File' y luego selecciona 'Save'.

Para poder correr el archivo que hemos escrito debemos primero empezar nuestro entorno de desarrollo siguiendo los pasos explicados en la parte 2.1.2 de este taller. Una vez que se abra la terminal debemos dirigirnos a la ruta donde están nuestros archivos. Para eso podemos ocupar el comando 'cd' que nos permite movernos a través de directorios.



Desde el símbolo de sistema, llamaremos al interpretador de Python usando 'python hola\_mundo.py'

```
C:\Windows\system32\cmd.exe
(curso_de_python) D:\Home\teaching\semester_2\curso_de_python\talleres\taller_1>python hola_mundo.py
Hola mundo!
(curso_de_python) D:\Home\teaching\semester_2\curso_de_python\talleres\taller_1>
```

## 4. Saludando desde el simbolo de sistema

Recibir datos desde el símbolo del sistema es posible en python, gracias a la ayuda de **input()**, de esta manera obtendremos un dato de tipo **string**, la cual puede ser guardada en un variable, finalmente modificaremos **print** para imprimir esta.

```
nombre = input("Ingresa tu nombre: ")
print("Hola ", nombre)
```

```
C:\Windows\system32\cmd.exe
(intro_python) c:\Users\Diego\Desktop\curso_de_python>python amongus.py
Ingresa tu nombre: Amongus
Hola Amongus
(intro_python) c:\Users\Diego\Desktop\curso_de_python>_
```

## 5. Instalando librerías

Instalar librerías en Python es trivial gracias a herramientas como pip. Vamos a instalar un paquete popular en el sector de análisis de datos, utilizada para la visualización de datos conocida como “**matplotlib**” con ayuda del comando **pip install -U <nombre del paquete>** dentro de tu terminal o **pip3 install -U <nombre del paquete>** si tienes múltiples versiones.

## 6. Trabajando con Jupyter Notebooks in VS

Marcel Mauricio Moran Calderon  
marcel\_moran41@hotmail.com

Ariel Ramos  
ariel.ramos97@gmail.com